

\* NOTICES \*

JPO and INPI T are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.

2.\*\*\*\* shows the word which can not be translated.

3.In the drawings, any words are not translated.

---

CLAIMS

---

[Claim(s)]

[Claim 1] The printing server characterized by establishing a means to control so that said two or more threads perform simultaneously rendering processing of two or more of said applications by performing the exclusive operation of two or more of said applications in the printing server which has two or more applications and two or more threads.

[Claim 2] The printing server characterized by establishing a means to record accounting information in a printing server according to claim 1 in case rendering processing is carried out by said each thread, respectively.

[Claim 3] The program for making it function on a computer as a means to control so that two or more threads perform simultaneously rendering processing of two or more of said applications by performing the exclusive operation of two or more applications.

[Claim 4] The record medium which recorded the program for making it function on a computer as a means to control so that two or more threads perform simultaneously rendering processing of two or more of

said applications by performing the exclusive operation of two or more applications and in which computer reading is possible.

---

[Translation done.]

\* NOTICES \*

JPO and INPI T are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.

2.\*\*\*\* shows the word which can not be translated.

3.In the drawings, any words are not translated.

---

## DETAILED DESCRIPTION

---

[Detailed Description of the Invention]

[0001]

[Field of the Invention] This invention relates to the record medium which recorded the program which operates on printing servers, such as a computer for printing the document of the documentation management system which manages as a document the meeting of the data which constitute pages, such as image data and text data, and a documentation management system, and the printing server of those, and its program and in which computer reading is possible.

[0002]

[Description of the Prior Art] Printing servers, such as a personal computer (PC) which has connected two or more sets of printers conventionally, The documentation- management server which uses as a "document" the set of an image file or an application file (file which

applications, such as a word processor and spreadsheet software, create), and manages it on a database and which is similarly PC etc., The document access which displayed the information in the above-mentioned printing server and a documentation-management server on the operator, and was accumulated into the documentation-management server, There is a system which has client terminal units, such as PC to which printing directions are made to perform the document accumulated into the documentation-management server and the data created, respectively by the printing server.

[0003] In order that two or more client terminal units may share two or more sets of printers on a network in such a system, The merit which performs printing control which supplies the printing request from each client terminal unit to each printer by the printing server Are unnecessary in the time and effort which installs the printer driver of each printer in a client terminal unit side, It is mentioned that it is not necessary to install the application for carrying out the rendering (changing a format of an application proper into the format which can be printed using the application) of the data etc. (For example, refer to JP,10-116161,A, JP,10-285324,A, JP,11-119945,A, JP,11-259258,A, JP,2000-35870,A, JP,2001-117747,A, and JP,2001-243034,A)

[0004]

[Problem(s) to be Solved by the Invention] However, by the above conventional systems, the problem that the burden concerning a printing server side was large was in the carrier beam case about the print job simultaneously from two or more client terminal units by the printing server side. Moreover, since print jobs were processed one by one, there was also a fault that two or more application processes could not be simultaneously performed in the rendering using application.

[0005] Then, if two or more queues are formed in a printing server side and two or more kinds of applications are started using each of that queue in order to solve an above-mentioned problem, the problem that two or more processes cannot be started depending on application will newly arise. It is necessary to control starting of application. This is because there are what can perform simultaneous starting according to the structure of application, and a thing which is not made. For example, although starting of two or more processes can do application A, Application B is being unable to perform starting of two or more processes (it being because there being constraint that a post process is impossible, for example, when two or more process starting is carried out for a rendering).

[0006] Concretely, although two or more process starting is possible for the WORD (Microsoft Word: trademark) of Microsoft Corp., Lotus 1-2-3 (Lotus 1-2-3: trademark) of Lotus has the problem of remaining in memory, without completing application, when two or more processes are started. It is made in order that this invention may solve the above-mentioned technical problem, and it aims at enabling it to perform efficiently simultaneous rendering processing of two or more application files.

[0007]

[Means for Solving the Problem] This invention offers the printing server of the following (1) and (2) in order to attain the above-mentioned object.

(1) The printing server which established a means to control so that two or more above-mentioned threads perform simultaneously rendering processing of two or more above-mentioned applications by performing the exclusive operation of two or more above-mentioned applications in the printing server which has two or more applications and two or more

threads.

(2) The printing server which established a means to record accounting information in the printing server of (1) in case rendering processing is carried out by each above-mentioned thread, respectively.

[0008] Furthermore, the following program of (3) and the record medium of (4) are also offered.

(3) The program for making it function on a computer as a means to control so that two or more threads perform simultaneously rendering processing of two or more above-mentioned applications by performing the exclusive operation of two or more applications.

(4) The record medium which recorded the program for making it function on a computer as a means to control so that two or more threads perform simultaneously rendering processing of two or more above-mentioned applications by performing the exclusive operation of two or more applications and in which computer reading is possible.

[0009]

[Embodiment of the Invention] Hereafter, the operation gestalt of this invention is concretely explained based on a drawing. Drawing 1 is the block diagram showing the internal configuration of the printing server which is 1 operation gestalt of this invention. Drawing 2 is the block diagram showing an example of the system configuration equipped with the printing server which is 1 operation gestalt of this invention. As shown in drawing 2, this system is the same with the printing servers 1, such as PC, and two or more sets (printer A- C) of Printers 3a- 3c are connected with two or more client terminal units (client A- C) 2a- 2c, such as PC, through the network.

[0010] The printing server 1 builds in the microcomputer which consists of CPU, a ROM, RAM, etc., realizes the function of each function part as shown in drawing 1 which that microcomputer mentions later, and

performs printing control concerning this invention. Then, the program for making it function as a means to control so that two or more threads perform simultaneously rendering processing of two or more of said applications is stored by performing the exclusive operation of two or more applications to ROM of the above-mentioned microcomputer etc. Moreover, the above-mentioned program is recorded on record media, such as an optical disk and a floppy disk, and you may make it install through the record medium.

[0011] That is, the printing server 1 has two or more applications and two or more threads, and achieves the function of a means to control so that two or more above-mentioned threads perform simultaneously rendering processing of two or more above-mentioned applications, by performing the exclusive operation of two or more above-mentioned applications. Moreover, in case rendering processing is carried out by each above-mentioned thread, respectively, the function of a means to record accounting information is also achieved.

[0012] Similarly each client terminal units 2a-2c contain a microcomputer, and require a printing request of the printing server 1. Each printers 3a-3c are airline printers, such as a laser beam printer, and print the print data sent from the printing server 1. As shown in drawing 1, the printing server 1 interior is equipped with the rendering application executive process 14 which manages two or more application groups for the renderings of the job of the printing request from each printer driver (printer driver A-C) 13a-13c and each client terminal units 2a-2c of each printers 3a-3c.

[0013] The printing server 1 is stored in each queues 11a-11c if the printing executive process 10 receives the request of printing from each client terminal units 2a-2c. The printing request is the file of the format of a job. In the printing server 1, it has the queues (queue A-C) 11a-11c

to each of each usable printers 3a- 3c, and the job in each queue 11a - 11c is supervised by each threads (thread A- C) 12a- 12c of the printing executive process 10. The application file (file created by software, such as a word processor and a spreadsheet) which should be printed is contained in a job.

[0014] Drawing 3 is the explanatory view showing the content of data stored in a job. The "job information" which becomes data of a job from a client terminal unit name, input time, the count of a retry, etc., The "printer name" which is the identifiers (for example, "printer A" etc.) of a printer which should be printed, The "after- treatment information" which is "printed information", such as a number of sets and a paper size, and the information about the processing performed after printing of punch, a staple, etc., The "application information" which is the application identification information (an extension, starting option, etc.) which shows the application which carries out a rendering. It consists of an "application data" which is live data (the data to print, the content of the application file) of application. Although this job is put into a queue, a file or the data on memory is sufficient as the job itself. It is dependent on the management method of a queue in what kind of format a job is managed. With this operation gestalt, explanation is advanced on the assumption that a queue is managed by the file.

[0015] Next, the function of this printing server 1 is explained in detail. In this printing server 1, the job of the printing request inputted from each client terminal units 2a- 2c is independently processed to every printer 3a - 3c. Therefore, two or more queues 11a- 11c are possessed, and the job about one set of a printer is stored in each queue 11a- 11c. Each queue 11a- 11c is supervised by the threads (parallel- processing process) 12a- 12c started from the printing executive process 10. Therefore, the number of each threads 12a- 12c and each queues

11a-11c is set to 1:1.

[0016] In case each threads 12a-12c process the job in each queue 11a-11c, they ask whether a rendering is possible to the rendering application executive process 14. The inquiry is judged with reference to the application starting managed table 15 based on the content. Various kinds of rendering applications which carry out the rendering of the job are also stored in the rendering application executive process 14.

drawing 1 -- as an example -- rendering application (application A) 14a of a word processor A, rendering application (application B) 14b of a word processor B, and the rendering application (application C) 14c of a spreadsheet -- rendering application (application D) 14d of c and CAD is stored. Although each threads 12a-12c perform a rendering immediately, when application is using it and more than one cannot be started in other processes, in the case of the job in which a rendering is possible, the reclosing of the applicable job is carried out to a queue, and they process the following job.

[0017] Next, the printing control processing concerning claim 1 of this invention in this printing server 1 is explained. The printing control processing is processing until it goes into the queue of the printing server 1, and the job by which the printing request was carried out from the client terminal unit performs a rendering, is passed to a printer driver and printed by the printer. To the printing server 1, each client terminal units 2a-2c send the job of the format shown in drawing 3, and carry out a printing request. The printing executive process 10 of the printing server 1 interior receives the job. The printing executive process 10 chooses the queue which supplies a job from Queues 11a-11c based on the "printer name" in a job (decision), and stores a job in the queue in the form of a file.

[0018] Drawing 5 is flow chart drawing showing the processing of the

printing executive process 10 shown in drawing 1. If the printing executive process 10 starts a thread at step ("S" shows among drawing) 1, judges whether there was any printing request from a client terminal unit at step 2 and has a printing request, it acquires the job of a printing request at step 3, chooses a queue from the printer name of the job at step 4, creates and supplies a job file (the file of a job) to the selected queue at step 5, and will return to step 2.

[0019] Furthermore, the printing executive process 10 starts Threads 12a- 12c to every queue 11a - 11c. If a taken charge queue is supervised respectively and a job is supplied in a queue, refer to the application starting managed table 15 for each threads 12a- 12c.

[0020] Drawing 4 is the explanatory view showing the content of data of the application starting managed table 15. The application starting information for the application stored in the rendering APURISHON executive process 14 is stored in this application starting managed table 15. To each application starting information, beforehand, if two or more startings are good, it memorizes "two or more startings are good", and if the corresponding rendering application cannot be two or more started, it does not memorize "two or more startings are good." Moreover, if the corresponding rendering application has not started and "un- starting" is [ be / it ] under starting, "under starting" is memorized, respectively.

[0021] Before Threads 12a- 12c start the rendering application corresponding to a job, refer to the rendering application starting information on rendering application that it corresponds in the application starting managed table 15 for them. If "two or more startings are good" is memorized by rendering application starting information, the rendering application will be started and a rendering will be made to perform irrespective of the existence of starting of the corresponding rendering application, without updating rendering application starting

information.

[0022] If "un-starting" is memorized by rendering application starting information, will update rendering application starting information "during starting", will start the corresponding rendering application, a rendering will be made to perform, and rendering application starting information will be updated at the time of termination of a rendering "un-starting." "Under starting" is memorized by rendering application starting information, and if "two or more startings are good" is not memorized, it polls, making processing stand by without performing starting of the corresponding rendering application until rendering application starting information is "un-starting."

[0023] As the above-mentioned standby, when rendering application is using [ be / it ] it, Threads 12a- 12c lower the priority of a queue for the job, and pass the following job to rendering application. In order to lower the priority of a queue, after putting a job into the backmost row of a queue, or making job data shunt and performing the following job, it is good to process performing the job again etc.

[0024] Those data are taken out to ejection and rendering application, and Threads 12a- 12c take out printing directions from a job for "application information" and an "application data" to delivery and rendering application, when starting rendering application. At this time, "printed information" and the "after-treatment information" within a job are set as printer drivers 13a- 13c.

[0025] The rendering applications 14a- 14d are application groups changed into the format which can be printed from a format of an application proper, and consist of software, such as a word processor, a spreadsheet, and CAD. Those rendering applications use the print facility which oneself has, pass it to the printer drivers 13a- 13c directed after changing the application data sent from Threads 12a- 12c, and

issue printing directions. In addition, when the rendering application for which two or more startings are improper is using it, that for which it can substitute out of usable rendering application is selected, and you may make it make a rendering perform to the alternative rendering application.

[0026] Print data are made to send and print from the rendering applications 14a- 14d to the printers 3a- 3c by which the carrier beam printer drivers 13a- 13c correspond printing directions, respectively. Or you may make it complete printing processing by storing data in the printing spooler of OS with an operating system (OS).

[0027] Thus, the job over each printer can be made to be able to become independent, and the processing effectiveness when performing simultaneously the rendering by two or more application files can be improved. In addition, a rendering is processed by order with a request when the number of queues is one like the conventional printing server. In this case, it is not concerned with whether the printing request was advanced by which printer, and since activation of a print job is processed in the sequence supplied to the queue, printing processing cannot be performed efficiently.

[0028] Next, in this printing server 1, printing processing can also be charged including a rendering with the job of every printer 3a - 3c, and the use count of the rendering application used for the rendering. For example, when printing by the rendering is successful, it becomes realizable by saving accounting information, such as printing number of sheets and a use count of printing conditions (a color, monochrome, etc.) and rendering application. The accounting information is good to make it record on memory, such as RAM which omitted the graphic display in the printing server 1.

[0029] Drawing 6 is flow chart drawing showing the printing control in

this printing server 1. A queue is supervised at step ("S" shows among drawing) 11, if it judges whether there is any job by the printing request from a client terminal unit and is at step 12, the rendering application of the rendering point will be chosen at step 13, and if it judges whether it is under [ activity ] \*\*\*\*\* and is [ be / it ] under activity by other threads at step 14, the priority of a job is lowered at step 19 and it returns to step 11. If it judges that it is not under activity by other threads at step 14, printing conditions (printed information, after-treatment information) will be set as a printing driver at step 15. Start rendering application at step 16 and conversion and printing are performed. If it judges whether printing was successful and does not succeed at step 17, error processing is carried out at step 20, this processing is ended, if it is a printing success, accounting information is recorded and saved at step 18, and this processing is ended.

[0030] above-mentioned step 13- it may be made to perform the next processing by processing of 16 and 19. Before Threads 12a- 12c start the rendering application corresponding to a job, refer to the rendering application starting information on rendering application that it corresponds in the application starting managed table 15 for them. the \*\* which will not update rendering application starting information irrespective of the existence of starting of the corresponding rendering application if "two or more startings are good" is memorized by rendering application starting information -- the -- rendering application is carried out and a rendering is made to perform

[0031] If "un-starting" is memorized by rendering application starting information, will update rendering application starting information "during starting", will start the corresponding rendering application, a rendering will be made to perform, and rendering application starting information will be updated at the time of termination of a rendering

"un-starting." "Under starting" is memorized by rendering application starting information, and if "two or more startings are good" is not memorized, it polls, making processing stand by without performing starting of the corresponding rendering application until rendering application starting information is "un-starting."

[0032] Thus, the toll of the application used can be distributed to a client terminal unit by charging also to the rendering of not only a printing result but rendering application. Like an application service provider (ASP), this is effective to set up a service price also including the toll of rendering application, in case it is offered as one service including printing and a rendering.

[0033] Next, other processings in this printing server 1 are explained. In this printing server 1, if scheduling of job processing is performed, a time zone with few server loads, such as a holiday, can be printed at night. In that case, a job-processing rule is prepared in each thread 12a- 12c of every. Each threads 12a- 12c perform printing processing based on a job-processing rule. For example, if there is a rule of processing at night, it will print by setting as night the time amount which processes the job in queue 11a - 11c. Moreover, when any one set or two or more sets cannot use it about Printers 3a- 3c, it may be made to carry out an alternative output at other printers. In that case, a vacant printer is used by delivering a job among thread 12a - 12c.

[0034] In the printing server 1, Threads 12a- 12c are performing control which takes out a job from Queues 11a- 11c, respectively. Then, activation of a print job is controllable by performing propriety of the ejection of the job from each queue 11a- 11c, and setting out of the operating time to each threads 12a- 12c. Refer to the thread activation definition file for each threads 12a- 12c in the actuation.

[0035] Drawing 7 is the explanatory view showing a format of a thread

activation definition file. The definition of the execution time and the actuation at the time of a printer halt are described by the thread activation definition file, and there is setting-out information on items, such as "demand processing", "processing classification", the "night processing time", the "weekend processing time", and "alternative processing", in it. Before a rendering, Threads 12a- 12c read setting-out information from a thread activation definition file, and opt for the processing when the ability not to use activation timing or Printers 3a- 3c from the setting-out information.

[0036] Drawing 8 is flow chart drawing showing processing actuation of the threads 12a- 12c of drawing 1. If Threads 12a- 12c are started, the thread will read and acquire a thread information definition file at step ("S" shows among drawing) 31, and will acquire the information on delay processing, and the information on alternative processing. It judges whether it is demand processing at step 32, if it is not demand processing, it will shift to processing of flow chart drawing shown in drawing 6, if it is demand processing, it judges whether it is the time of day which can be processed at step 33, and if it is not the time of day which can be processed, sleep processing will be carried out at step 38, and it will return to step 32.

[0037] If judge whether the printer which corresponds at step 34 is usable when judging it as the time of day which can be processed at step 33, it shifts to processing of flow chart drawing shown in drawing 6 if , it judges whether alternative printing processing is performed at step 35 if , and alternative printing is not performed, error processing (printer halt) will be performed at step 39, and this processing will be ended. If alternative printing is performed at step 35, a print facility judges a \*\*\*\*\* enough at step 36, if a print facility is not enough, error processing (lack of printer ability) will be performed at step 39, and this

processing will be ended.

[0038] If it judges that a print facility is enough at step 36, a job will be updated at step 37, that job will be registered into an applicable queue, and this processing will be ended. In above-mentioned processing, when the record of "demand processing" of thread activation definition FAIRU is TRUE, it shifts to decision whether it can process or not and, in FALSE, shifts to processing of drawing 6. Moreover, when the record of "processing classification" is "night" and "holiday" processing, it refers to system time of day, it judges whether it is the time of day when the present time of day should perform printing processing, and if it is not the time of day which should perform printing processing, sleep processing for which it waits until it reaches at processing time of day, carrying out a wait will be performed. Drawing 6 will be processed if delay processing is not required.

[0039] Next, the activity propriety of Printers 3a- 3c is investigated. The activity propriety of Printers 3a- 3c is judged with means, such as asking a printer driver the status of Printers 3a- 3c, and when either of the printers 3a- 3c cannot use it, the record of "alternative processing" of a thread activation definition file is referred to. When the record of "alternative processing" is FALSE, it ends in the error under printer halt. If the record of "alternative processing" is TRUE, the processing facility of a printer will be checked. For example, the printer for which they will substitute if it is required for options, such as punch and a staple, whether it is the need confirms whether to have those functions. If the alternative with other printers is not possible for the corresponding job, it will end in the error in short of alternative printer ability. Moreover, a print job will be updated if the alternative with other printers is possible for the corresponding job.

[0040] Processing is ended by supplying to the queue of the printer for

which the printer name of the printer which substitutes for a "printer name" among the jobs shown in [drawing 3](#) is rewritten and substituted, namely, entrusting a job to other threads. Thus, a server load can be distributed by printing a time zone with few loads of a printing server. Moreover, when a printer is maintaining, a job is stopped, and it becomes possible to prevent generating of an excessive rendering. Furthermore, when one of the printers connected is maintaining during failure (paper jam etc.), relief of the latency time can be aimed at by outputting to other printers.

[0041]

[Effect of the Invention] It can make it possible to perform efficiently simultaneous rendering processing of two or more application files according to the printing server, program, and record medium of this invention, as explained above.

---

[Translation done.]

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2003-323273

(43)Date of publication of application : 14.11.2003

(51)Int.Cl.

G06F 3/12

B41J 29/38

G06F 9/46

G06F 15/00

(21)Application number : 2002-128485

(71)Applicant : RICOH CO LTD

(22)Date of filing : 30.04.2002

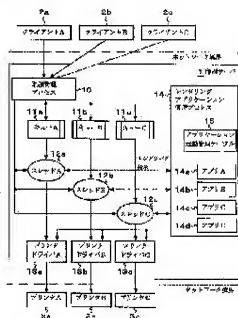
(72)Inventor : II YASUHIRO

## (54) PRINT SERVER, PROGRAM AND RECORDING MEDIUM

### (57)Abstract:

**PROBLEM TO BE SOLVED:** To efficiently perform simultaneous rendering processing of a plurality of application files.

**SOLUTION:** Threads 12a to 12c, before starting up a rendering application corresponding to a job, refer to an application start-up management table 15, if corresponding rendering application start-up information includes a message of 'start-up of a plurality of the applications is possible' or 'not yet started up', start up the rendering application to render, and if a message of 'in start-up' is stored instead of 'start-up of the plurality of the applications is possible', do not start up the corresponding rendering application and poll while holding the processing on standby until the rendering application start-up information is turned, indicating 'not yet started up'.



## LEGAL STATUS

[Date of request for examination]

22.10.2004

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2003-323273

(P2003-323273A)

(43) 公開日 平成15年11月14日 (2003.11.14)

(51) Int.Cl. <sup>7</sup>	識別記号	F I	データベース <sup>*</sup> (参考)
G 0 6 F	3/12	G 0 6 F	3/12 C 2 C 0 6 1
			B 5 B 0 2 1
B 4 1 J	29/38	B 4 1 J	29/38 Z 5 B 0 8 5
G 0 6 F	9/46	G 0 6 F	9/46 3 4 0 B 5 B 0 9 8
	15/00		15/00 3 1 0 A
	3 1 0	審査請求	未請求 請求項の数 4 O L (全 8 頁)

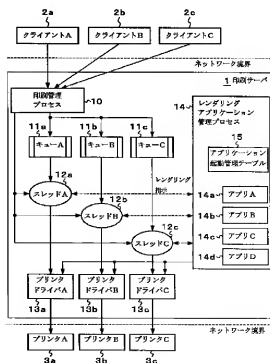
(21) 出願番号	特願2002-128485(P2002-128485)	(71) 出願人	000006747 株式会社リコー 東京都大田区中馬込 1 丁目 3 番 6 号
(22) 出願日	平成14年 4 月 30 日 (2002. 4. 30)	(72) 発明者	伊井 泰洋 東京都大田区中馬込 1 丁目 3 番 6 号 株式 会社リコー内
		(74) 代理人	100080931 弁理士 大 都 敬 F ターム(参考) 20061 AP01 HQ12 HQ17 HR07 HX10 5B021 AA01 B301 CC04 NN00 5B085 AC04 BA06 BE07 BG02 BG04 BG07 5B098 AA08 GA05 GC01 GD02 GD15

(54) 【発明の名称】 印刷サーバとプログラムと記録媒体

(57) 【要約】

【課題】 複数のアプリケーションファイルの同時レンダリング処理を効率良く行えるようにする。

【解決手段】 スレッド12a~12cは、ジョブに対応するレンダリングアプリケーションを起動させる前にアプリケーション起動管理テーブル15を参照し、該当するレンダリングアプリケーション起動情報に「複数起動可」又は「未起動」が有ればそのレンダリングアプリケーションさせてレンダリングを行わせ、「複数起動可」が記憶されてなくて「起動中」があれば、該当するレンダリングアプリケーションの起動は行わずに、レンダリングアプリケーション起動情報が「未起動」になるまで処理を待機させつつポーリングを行う。



## 【特許請求の範囲】

【請求項1】 複数のアプリケーションと複数のスレッドを有する印刷サーバにおいて、前記複数のアプリケーションの排他処理を行うことにより前記複数のアプリケーションのレンダリング処理を前記複数のスレッドにより同時に行うように制御する手段を設けたことを特徴とする印刷サーバ。

【請求項2】 請求項1記載の印刷サーバにおいて、前記各スレッドによってそれぞれレンダリング処理をする際に課金情報を記録する手段を設けたことを特徴とする印刷サーバ。

【請求項3】 コンピュータに、複数のアプリケーションの排他処理を行うことにより前記複数のアプリケーションのレンダリング処理を複数のスレッドにより同時に行うように制御する手段として機能させるためのプログラム。

【請求項4】 コンピュータに、複数のアプリケーションの排他処理を行うことにより前記複数のアプリケーションのレンダリング処理を複数のスレッドにより同時に行うように制御する手段として機能させるためのプログラムを記録したコンピュータ読み取り可能な記録媒体。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】 この発明は、画像データやテキストデータなどのページを構成するデータの集まりを文書として管理する文書管理システム、文書管理システムの文書を印刷するためのコンピュータ等の印刷サーバとその印刷サーバ上で動作するプログラムとそのプログラムを記録したコンピュータ読み取り可能な記録媒体とに関する。

## 【0002】

【従来の技術】 従来、複数台のプリンタを接続しているパーソナルコンピュータ（PC）等の印刷サーバと、データベース上で画像ファイルもしくはアプリケーションファイル（ワープロ、表計算ソフト等のアプリケーションが作成するファイル）の集合を「文書」として管理する同じPC等の文書管理サーバと、上記印刷サーバ及び文書管理サーバ内の情報を操作者に表示し、文書管理サーバ内に蓄積された文書閲覧や、文書管理サーバ内に蓄積された文書やそれで作成したデータを印刷サーバによって印刷指示を行わせるPC等のクライアント端末装置を有するシステムがある。

【0003】 このようなシステムにおいて、複数台のクライアント端末装置がネットワーク上で複数台のプリンタを共有するため、印刷サーバによって各クライアント端末装置からの印刷リクエストを各プリンタへ配給する印刷制御を行うメトリックは、クライアント端末装置側に各プリンタのプリンタドライバをインストールする手間が必要なこと、データをレンダリング（アプリケーション固有のフォーマットをそのアプリケーションを用い

て印刷可能なフォーマットに変換すること）するためのアプリケーションをインストールしなくて済むことなどが挙げられる。（例えば、特開平10-116161号公報、特開平10-285324号公報、特開平11-119945号公報、特開平11-259258号公報、特開2000-35870号公報、特開2001-117747号公報、特開2001-243034号公報参照）

## 【0004】

【発明が解決しようとする課題】 しかしながら、上述のような従来のシステムでは、印刷サーバ側で複数台のクライアント端末装置から同時に印刷ジョブを受けた場合に、印刷サーバ側に掛かる負担が大ききという問題があった。また、印刷ジョブをひとつひとつ処理するので、アプリケーションを使ったレンダリングにおいて同時に複数のアプリケーション処理ができないという欠点もあった。

【0005】 そこで、上述の問題を解消するために、印刷サーバ側に複数のキューを設け、その各キューを用いて複数種類のアプリケーションを起動するようにすると、アプリケーションによっては複数プロセスが起動できないという問題が新たに生じる。アプリケーションの起動を制御する必要がある。これはアプリケーションの構造によって同時起動ができるものとできないものがあるせいである。例えば、アプリケーションAは複数プロセスの起動ができるが、アプリケーションBは複数プロセスの起動ができない（レンダリングのために複数プロセス起動すると、例えば終了処理ができないという制約があるためである）ということである。

【0006】 具体的に、例えば、マイクロソフト社のワード（Microsoft Word：登録商標）は複数プロセス起動が可能だが、ロータス社のロータス1-2-3（Lotus 1-2-3：登録商標）は複数プロセスの起動を行うと、アプリケーションが終了せずにメモリに残るという問題がある。この発明は上記の課題を解決するためになされたものであり、複数のアプリケーションファイルの同時レンダリング処理を効率良く行うようにすることを目的とする。

## 【0007】

【課題を解決するための手段】 この発明は上記の目的を達成するため、次の（1）と（2）の印刷サーバを提供する。

（1） 複数のアプリケーションと複数のスレッドを有する印刷サーバにおいて、上記複数のアプリケーションの排他処理を行うことにより上記複数のアプリケーションのレンダリング処理を上記複数のスレッドにより同時に行うように制御する手段を設けた印刷サーバ。

（2） （1）の印刷サーバにおいて、上記各スレッドによってそれぞれレンダリング処理をする際に課金情報を記録する手段を設けた印刷サーバ。

【0008】さらに、次の(3)のプログラム及び(4)の記録媒体も提供する。

(3) コンピュータに、複数のアプリケーションの排他処理を行うことにより上記複数のアプリケーションのレンダリング処理を複数のスレッドにより同時に行うように制御する手段として機能させるためのプログラム。

(4) コンピュータに、複数のアプリケーションの排他処理を行うことにより上記複数のアプリケーションのレンダリング処理を複数のスレッドにより同時に行うように制御する手段として機能させるためのプログラムを記録したコンピュータ読み取り可能な記録媒体。

【0009】

【発明の実施の形態】以下、この発明の実施形態を図面に基づいて具体的に説明する。図1は、この発明の一実施形態である印刷サーバの内部構成を示すブロック図である。図2は、この発明の一実施形態である印刷サーバを備えたシステム構成の一例を示すブロック図である。図2に示すように、このシステムは、P C等の印刷サーバ1と、同じくP C等の複数のクライアント端末装置(クライアントA～C) 2 a～2 cと複数のプリンタ(プリンタA～C) 3 a～3 cがネットワークを介して接続されている。

【0010】印刷サーバ1は、CPU、ROM及びRAM等からなるマイクロコンピュータを内蔵し、そのマイクロコンピュータが後述する図1に示すような各機能部の機能を実現し、この発明に係る印刷制御を行う。そこで、上記マイクロコンピュータのROM等に複数のアプリケーションの排他処理を行うことにより前記複数のアプリケーションのレンダリング処理を複数のスレッドにより同時に行うように制御する手段として機能させるためのプログラムを格納する。また、上記プログラムを光ディスクやフロッピーディスク等の記録媒体に記録し、その記録媒体を介してインストールするようにしても良い。

【0011】すなわち、印刷サーバ1が、複数のアプリケーションと複数のスレッドを有し、上記複数のアプリケーションの排他処理を行うことにより上記複数のアプリケーションのレンダリング処理を上記複数のスレッドにより同時に行うように制御する手段の機能を果たす。また、上記各スレッドによってそれぞれレンダリング処理をする際に課金情報を記録する手段の機能も果たす。

【0012】各クライアント端末装置2 a～2 cは、同じくマイクロコンピュータを内蔵し、印刷サーバ1へ印刷リクエストを要求する。各プリンタ3 a～3 cは、レーザプリンタ等の印刷装置であり、印刷サーバ1から送られる印刷データを印刷する。図1に示すように、印刷サーバ1内部には、各プリンタ3 a～3 cのそれぞれのプリンタドライバ(プリンタドライバA～C) 1 3 a～1 3 cと、各クライアント端末装置2 a～2 cからの印刷リクエストのジョブのレンダリング用の複数のアプリ

ケーション群を管理するレンダリングアプリケーション管理プロセス1 4とを備えている。

【0013】印刷サーバ1は、印刷管理プロセス1 0が各クライアント端末装置2 a～2 cから印刷のリクエストを受けると、各キュー1 1 a～1 1 cに格納する。その印刷リクエストは、ジョブという形式のファイルである。印刷サーバ1内には、使用可能な各プリンタ3 a～3 cのそれぞれに対するキュー(キューA～C) 1 1 a～1 1 cを備え、印刷管理プロセス1 0の各スレッド(スレッドA～C) 1 2 a～1 2 cによってそれぞれのキュー1 1 a～1 1 c内のジョブを監視する。ジョブには、印刷すべきアプリケーションファイル(ワープロ、表計算等のソフトウェアで作成されたファイル)が含まれる。

【0014】図3は、ジョブに格納されるデータ内容を示す説明図である。ジョブのデータには、クライアント端末装置名、入力日時、リトリア回数等からなる「ジョブ情報」と、印刷すべきプリンタの識別子(例えば「プリンタA」等)である「プリンタ名」と、印刷部数、用紙サイズ等の「印刷情報」と、パンチ、ステープル等の印刷後に行う処理に関する情報である「後処理情報」と、レンダリングするアプリケーションを示すアプリケーション識別情報(拡張子、起動オプション等)である「アプリケーション情報」と、アプリケーションの実データ(印刷するデータ、アプリケーションファイルの内容)である「アプリケーションデータ」とからなる。このジョブはキュー1 1 a～1 1 cには1台のプリンタに関するジョブを格納する。それぞれのキュー1 1 a～1 1 cを、印刷管理プロセス1 0から起動されたスレッド(並行処理プロセス) 1 2 a～1 2 cによって監視する。したがって、各スレッド1 2 a～1 2 cと各キュー1 1 a～1 1 cの数は1:1になる。

【0015】次に、この印刷サーバ1の機能について詳しく説明する。この印刷サーバ1では、各クライアント端末装置2 a～2 cから入力される印刷リクエストのジョブの処理を、プリンタ3 a～3 c毎に独立して行う。そのために、複数のキュー1 1 a～1 1 cを所持し、それぞれのキュー1 1 a～1 1 cには1台のプリンタに関するジョブを格納する。それぞれのキュー1 1 a～1 1 cを、印刷管理プロセス1 0から起動されたスレッド(並行処理プロセス) 1 2 a～1 2 cによって監視する。したがって、各スレッド1 2 a～1 2 cと各キュー1 1 a～1 1 cの数は1:1になる。

【0016】各スレッド1 2 a～1 2 cは、それぞれのキュー1 1 a～1 1 c内のジョブを処理する際に、レンダリングアプリケーション管理プロセス1 4に対してレンダリング可能か否かを問い合わせる。その問い合わせは、アプリケーション起動管理テーブル1 5を参照し、その内容に基づいて判断する。レンダリングアプリケーション管理プロセス1 4には、ジョブをレンダリングする各種のレンダリングアプリケーションも格納してい

る。図1には、一例としてワープロAのレンダリングアプリケーション（アプリA）14a、ワープロBのレンダリングアプリケーション（アプリB）14b、表計算のレンダリングアプリケーション（アプリC）14c、CADのレンダリングアプリケーション（アプリD）14dを格納している。各スレッド12a~12cは、レンダリング可能なジョブの場合は、即時にレンダリングを実行させるが、他プロセスでアプリケーションが使用中で複数起動できない場合は、該当ジョブをキューに再投入して次のジョブを処理する。

【0017】次に、この印刷サーバ1におけるこの発明の請求項1に係る印刷制御処理について説明する。その印刷制御処理は、クライアント端末装置から印刷リクエストされたジョブが、印刷サーバ1のキューに入り、レンダリングを行ってプリンタドライバに渡され、プリンタによって印刷されるまでの処理である。各クライアント端末装置2a~2cは、印刷サーバ1に対して、図3に示したフォーマットのジョブを送って印刷リクエストする。そのジョブは、印刷サーバ1内部の印刷管理プロセス10が受け取る。印刷管理プロセス10は、ジョブの中の「プリンタ名」に基づいてキュー11a~11cからジョブを投入するキューを選択（決定）し、そのキューにジョブをファイルの形式で格納する。

【0018】図5は、図1に示した印刷管理プロセス10の処理を示すフローチャート図である。印刷管理プロセス10は、ステップ（図中「S」で示す）1でスレッドを起動し、ステップ2でクライアント端末装置からの印刷リクエストが有ったか否かを判断し、印刷リクエストが有ったら、ステップ3で印刷リクエストのジョブを取得し、ステップ4でそのジョブのプリンタ名からキューを選択し、ステップ5でその選択したキューにジョブファイル（ジョブのファイル）を作成して投入し、ステップ2へ戻る。

【0019】さらに、印刷管理プロセス10は、キュー11a~11c毎にスレッド12a~12cを起動する。各スレッド12a~12cは、各々担当分のキューを監視し、キューの中にジョブが投入されると、アプリケーション起動管理テーブル15を参照する。

【0020】図4は、アプリケーション起動管理テーブル15のデータ内容を説明する図である。このアプリケーション起動管理テーブル15には、レンダリングアプリケーション管理プロセス14に格納されたアプリケーション分のアプリケーション起動情報を格納する。各アプリケーション起動情報には、該当するレンダリングアプリケーションが予め複数起動可なら「複数起動可」を記憶し、複数起動不可なら「複数起動不可」を記憶しない。また、該当するレンダリングアプリケーションが未起動なら「未起動」を、起動中なら「起動中」をそれぞれ記憶する。

【0021】スレッド12a~12cは、ジョブに対応

するレンダリングアプリケーションを起動させる前にアプリケーション起動管理テーブル15内の該当するレンダリングアプリケーションのレンダリングアプリケーション起動情報を参照する。レンダリングアプリケーション起動情報に「複数起動可」が記憶されていれば、該当するレンダリングアプリケーションの起動の有無にかかわらず、レンダリングアプリケーション起動情報を更新せずに、そのレンダリングアプリケーションを起動させてレンダリングを行わせる。

10 【0022】レンダリングアプリケーション起動情報に「未起動」が記憶されていれば、レンダリングアプリケーション起動情報を「起動中」に更新し、該当するレンダリングアプリケーションを起動させてレンダリングを行わせ、レンダリングの終了時にレンダリングアプリケーション起動情報を「未起動」に更新する。レンダリングアプリケーション起動情報に「起動中」が記憶されており、「複数起動可」が記憶されていなければ、該当するレンダリングアプリケーションの起動は行わずに、レンダリングアプリケーション起動情報が「未起動」になるまで処理を待機させつつボーリングを行う。

20 【0023】上記待機として、スレッド12a~12cは、レンダリングアプリケーションが使用申込んだ場合、そのジョブをキューの優先度を下げて、次のジョブをレンダリングアプリケーションへ渡す。キューの優先度を下げるには、ジョブをキューの最後列に入れたり、あるいはジョブデータを持越させておき、次のジョブを行った後に再度そのジョブを行うなどの処理をする。

30 【0024】スレッド12a~12cは、レンダリングアプリケーションを起動させるとき、ジョブから「アプリケーション情報」と「アプリケーションデータ」を取り出し、レンダリングアプリケーションにそれらのデータを渡し、レンダリングアプリケーションに印刷指示を出す。このとき、ジョブ内の「印刷情報」「後処理情報」をプリンタドライバ13a~13cに設定する。

40 【0025】レンダリングアプリケーション14a~14dは、アプリケーション固有のフォーマットから印刷可能なフォーマットに変換するアプリケーション群であり、ワープロ、表計算、CADなどのソフトウェアからなる。それらのレンダリングアプリケーションは、自らの持つ印刷機能を利用し、スレッド12a~12cから送られたアプリケーションデータを変換した後に指示されたプリンタドライバ13a~13cへ渡して印刷指示を出す。なお、複数起動が不可のレンダリングアプリケーションが使用中の場合に、使用可能なレンダリングアプリケーションの中から代替できるものを選び出し、その代替レンダリングアプリケーションにレンダリングを行わせるようにしてもよい。

50 【0026】レンダリングアプリケーション14a~14dから印刷指示を受けたプリンタドライバ13a~1

3 cは、それぞれ対応するプリンタ3 a~3 cに対して印刷データを送付して印刷させる。あるいは、オペレーティングシステム（OS）によっては、OSの印刷スプーラにデータを蓄積することにより印刷処理を完了するようにしてもよい。

【0027】このようにして、それぞれのプリンタに対するジョブを独立させ、複数のアプリケーションファイルによるレンダリングを同時に行うときの処理効率を向上させることができる。なお、従来の印刷サーバのようにキューがひとつの場合には、レンダリングはリクエストがあった順に処理される。この場合、どのプリンタに印刷リクエストが出されたかに関わらず、印刷ジョブの実行はキューに投入した順序で処理されるので、印刷処理を効率よく行えない。

【0028】次に、この印刷サーバ1において、プリンタ3 a~3 c毎のジョブと、レンダリングに使用したレンダリングアプリケーションの使用回数により、印刷処理にレンダリングを含めて課金することもできる。例えば、レンダリングによる印刷が成功した場合に、印刷枚数、印刷条件（カラー、モノクロ等）とレンダリングアプリケーションの使用回数などの課金情報を保存することにより実現が可能となる。その課金情報は、印刷サーバ1内の図示を省略したRAM等のメモリに記録するようにするとよい。

【0029】図6は、この印刷サーバ1における印刷制御を示すフローチャート図である。ステップ（図中「S」で示す）11でキューを監視し、ステップ12でクライアント端末装置からの印刷リクエストによるジョブの有るか否かを判断し、有ればステップ13でレンダリング先のレンダリングアプリケーションを選択し、ステップ14で他のスレッドで使用中か否かを判断し、使用中ならステップ19でジョブの優先度を下げてステップ11へ戻る。ステップ14で他のスレッドで使用中でないと判断したら、ステップ15で印刷ドライバに印刷条件（印刷情報、後処理情報）を設定し、ステップ16でレンダリングアプリケーションを起動して変換と印刷を行い、ステップ17で印刷が成功したか否かを判断し、成功しなければステップ20でエラー処理してこの処理を終了し、印刷成功ならステップ18で課金情報を記録して保存してこの処理を終了する。

【0030】上記ステップ13~16及び19の処理で次の処理を行うようにしても良い。スレッド12 a~12 cは、ジョブに対応するレンダリングアプリケーションを起動させる前にアプリケーション起動管理テーブル15内の該当するレンダリングアプリケーションのレンダリングアプリケーション起動情報を参照する。レンダリングアプリケーション起動情報に「複数起動可」が記憶されていれば、該当するレンダリングアプリケーションの起動の有無にかかわらず、レンダリングアプリケーション起動情報を更新せずに、そのレンダリングアプリ

ケーションさせてレンダリングを行わせる。

【0031】レンダリングアプリケーション起動情報に「未起動」が記憶されていれば、レンダリングアプリケーション起動情報を「起動中」に更新し、該当するレンダリングアプリケーションを起動させてレンダリングを行わせ、レンダリングの終了時にレンダリングアプリケーション起動情報を「未起動」に更新する。レンダリングアプリケーション起動情報に「起動中」が記憶されており、「複数起動可」が記憶されていなければ、該当するレンダリングアプリケーションの起動は行わずに、レンダリングアプリケーション起動情報が「未起動」になるまで処理を待機させつつポーリングを行う。

【0032】このようにして、印刷結果のみならず、レンダリングアプリケーションのレンダリングに対しても課金を行うことにより、使用アプリケーションの使用料金をクライアント端末装置に分配することができる。これはアプリケーションサービスプロバイダ（ASP）のように、印刷とレンダリングを含めてひとつのサービスとして提供する際に、レンダリングアプリケーションの使用料金も含めたサービス価格の設定を行いたい場合に有効である。

【0033】次に、この印刷サーバ1における他の処理について説明する。この印刷サーバ1において、ジョブ処理のスケジューリングを行えば、夜間、休日等のサーバ負荷の少ない時間帯の印刷を行うことができる。その場合、各スレッド12 a~12 c毎にジョブ処理ルールを設ける。各スレッド12 a~12 cはジョブ処理ルールに基づいて印刷処理を行う。例えば、夜間処理のルールがあれば、キュー11 a~11 c内のジョブを処理する時間を夜間に設定して印刷を行う。また、プリンタ3 a~3 cについていずれか一台または複数台が使用不可の場合、他のプリンタに代替出力するようにしてもよい。その場合、スレッド12 a~12 c間でジョブの受け渡しをすることにより、空いているプリンタを使用する。

【0034】印刷サーバ1において、キュー11 a~11 cからジョブを取り出す制御を行っているのはそれぞれスレッド12 a~12 cである。そこで、各スレッド12 a~12 cに対してそれぞれのキュー11 a~11 cからのジョブの取り出しの可否、動作時間の設定を行うことにより、印刷ジョブの実行の制御を行うことができる。各スレッド12 a~12 cはその動作において、スレッド実行定義ファイル16を参照する。

【0035】図7は、スレッド実行定義ファイルのフォーマットを示す説明図である。そのスレッド実行定義ファイルには、実行時間の定義やプリンタ停止時の動作が記述されており、「即時処理」「処理種別」「夜間処理時間」「週末処理時間」「代替処理」等の項目の設定情報がある。スレッド12 a~12 cは、レンダリングの前にスレッド実行定義ファイルより設定情報を読み取

り、その設定情報から実行タイミングあるいはプリンタ3a~3cが使用できない場合の処理を決める。

【0036】図8は、図1のスレッド12a~12cの処理動作を示すフローチャート図である。スレッド12a~12cが起動されると、そのスレッドは、ステップ(図中「S」で示す)31でスレッド情報定義ファイルを読み込んで取得し、遅延処理の情報、代替処理の情報を得る。ステップ32で即時処理か否かを判断し、即時処理でなければ図6に示したフローチャート図の処理へ移行し、即時処理ならステップ33で処理可能時刻か否かを判断し、処理可能時刻でなければステップ38でスリープ処理してステップ32へ戻る。

【0037】ステップ33で処理可能時刻と判断したら、ステップ34で該当するプリンタが使用可能か否かを判断し、使用可能なら図6に示したフローチャート図の処理へ移行し、使用不可能ならステップ35で代替印刷処理を行うか否かを判断し、代替印刷を行わないならステップ39でエラー処理(プリンタ停止)を行ってこの処理を終了する。ステップ35で代替印刷を行うなら、ステップ36で印刷機能は充分か否かを判断し、印刷機能が十分でなければステップ39でエラー処理(プリンタ機能不足)を行ってこの処理を終了する。

【0038】ステップ36で印刷機能が十分と判断したら、ステップ37でジョブを更新して該当キューにそのジョブを登録して、この処理を終了する。上述の処理で、スレッド実行定義ファイルの「即時処理」のレコードがTRUEの場合は、処理可能か否かの判断へ移行し、FALSEの場合は図6の処理へ移行する。また、「処理種別」のレコードが「夜間」「休日」処理の場合は、システム時刻と照らし合せ、現在の時刻が印刷処理を行うべき時刻かを判断し、印刷処理を行うべき時刻でなければ、ウェイトしながら処理時刻に達するまで待つスリープ処理を行う。遅延処理が必要でなければ、図6の処理を行う。

【0039】次に、プリンタ3a~3cの使用可否を調査する。プリンタドライバにプリンタ3a~3cのステータスを問い合わせるなどの手段でプリンタ3a~3cの使用可否を判断し、プリンタ3a~3cのいずれかが使用不可の場合は、スレッド実行定義ファイルの「代替処理」のレコードを参照する。「代替処理」のレコードがFALSEの場合は、プリンタ停止中のエラーで終了する。「代替処理」のレコードがTRUEであれば、プリンタの処理機能のチェックを行う。例えば、パンチ、ステープルなどの付加機能が必要か否か、それが必要であれば代替するプリンタがそれらの機能を有するかのチェックを行う。該当するジョブが他のプリンタで代替可

能でなければ、代替プリンタ機能不足のエラーで終了する。また、該当するジョブが他のプリンタで代替可能であれば印刷ジョブの更新を行う。

【0040】図3に示したジョブのうち、「プリンタ名」を代替するプリンタのプリンタ名に書き換えて、代替するプリンタのキューに投入する、すなわち、他のスレッドにジョブを委託することにより処理を終了する。このようにして、印刷サーバの負荷の少ない時間帯の印刷を行うことによってサーバ負荷を分散することができる。また、プリンタがメンテナンス中の場合などにジョブを停止させておき、余計なレンダリングの発生を防止することが可能になる。さらに、接続されているプリンタのひとつが故障中(紙づまり等)あるいはメンテナンス中の場合に、他のプリンタに出力することにより、待ち時間の軽減が図れる。

#### 【0041】

【発明の効果】以上説明してきたように、この発明の印刷サーバとプログラムと記録媒体によれば、複数のアプリケーションファイルの同時レンダリング処理を効率良く行えるようにすることが可能になる。

#### 【図面の簡単な説明】

【図1】この発明の一実施形態である印刷サーバの内部構成を示すブロック図である。

【図2】この発明の一実施形態である印刷サーバを備えたシステム構成の一例を示すブロック図である。

【図3】ジョブに格納されるデータ内容を示す説明図である。

【図4】図1に示したアプリケーション起動管理テーブル15のデータ内容を示す説明図である。

【図5】図1に示した印刷管理プロセス10の処理を示すフローチャート図である。

【図6】この印刷サーバ1における印刷制御を示すフローチャート図である。

【図7】スレッド実行定義ファイルのフォーマットを示す説明図である。

【図8】図1のスレッド12a~12cの処理動作を示すフローチャート図である。

#### 【符号の説明】

1：印刷サーバ 2a~2c：クライアント端末装置

3a~3c：プリンタ 10：印刷管理プロセス

11a~11c：キュー 12a~12c：スレッド

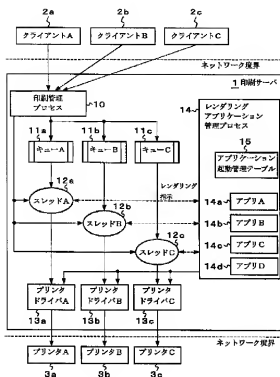
13a~13c：プリンタドライバ

14：レンダリングアプリケーション管理プロセス

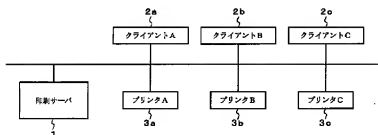
14a~14d：レンダリングアプリケーション

15：アプリケーション起動管理テーブル

【図1】



【図2】



【図3】

データ型	項目名	説明
Boolean	印刷処理	TRUEの場合は即時実行、FALSEの場合は遅延の処理
String	印刷時刻	"後夜" "深夜" "C/D延長時間"などの印刷タイミング
Time	印刷開始時間	印刷時刻が"後夜"の最初の印刷開始時間
Time	印刷終了時間	印刷時刻が"深夜"の最後の印刷終了時間
Boolean	印刷処理	TRUEの場合は遅延の処理、FALSEの場合は即時実行

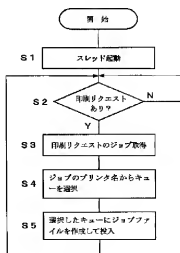
【図3】

ジョブ情報	クライアント名、入力日時、リソース情報
プリンタ名	印刷すべきプリンタの識別子 (プリンタIDなど)
印刷情報	印刷の種別、印刷サイズ
印刷処理情報	パンチ、ステープルなどの印刷処理情報
アプリケーション情報	アプリケーション識別情報 (拡張子、起動オプション)
アプリケーションデータ	アプリケーションの拡張子 (アプリケーションファイルの内部)

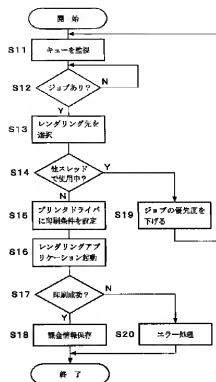
【図4】

アプリケーション管理テーブル
レンダリングアプリケーションA (印刷情報) 印刷中/未印刷/印刷完了
レンダリングアプリケーションB (印刷情報) 印刷中/未印刷/印刷完了
レンダリングアプリケーションC (印刷情報) 印刷中/未印刷/印刷完了
レンダリングアプリケーションD (印刷情報) 印刷中/未印刷/印刷完了

【図5】



【図6】



【図8】

